Package: QuantileModels (via r-universe)

October 11, 2026

Title Estimation of Different Quantile Related Models

Version 1.0.0

Description Estimation of different quantile models, at the moment only Conditional autoregressive value at risk (CAViaR) proposed by Engle & Manganelli (2004) <doi:10.1198/073500104000000370> with also the specification proposed in Huang et al. (2009) <doi:10.1016/j.eneco.2008.12.006> and it's multivariate extension, Multi-variate multi-quantile CAViaR (MVMQ-CAViaR) proposed by White et al. (2015) <doi:10.1016/j.jeconom.2015.02.004> are available, however, in further updates, other models and extensions will be included.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, nloptr, quantreg, numDeriv, xts, zoo, ufRisk, GenSA

Suggests knitr, rmarkdown,

VignetteBuilder knitr

Depends R (>= 3.5)

LazyData true

Config/roxygen2/version 8.0.0

Language en-US

Config/pak/sysreqs cmake

Repository https://chjoca.r-universe.dev

Date/Publication 2026-07-12 15:45:02 UTC

RemoteUrl https://github.com/chjoca/quantilemodels

RemoteRef HEAD

RemoteSha dd269dc9a0e0291d1ad5db94aaac296ccb39dd61

Contents

CAViaR 2
dataCAViaR 5
MVMQ 5
MVMQ_CAViaR 6
plot.CAViaR_estim 8
plot.MVMQ_CAViaR 8
print.CAViaR_estim 9
print.MVMQ_CAViaR 10
summary.CAViaR_estim 10
summary.MVMQ_CAViaR 11
wti_data 11

Index **12**

`CAViaR` *CAViaR model estimation*

Description

This function allows the estimation for a general order the caviar model proposed by Engle & Manganelli (2004).

Usage

```
CAViaR(
  Y,
  p = 1,
  q = 1,
  model.type = "SAV",
  tau = 0.05,
  band.hs = FALSE,
  jac.method = "Richardson",
  jac.options = list(),
  sign.level = 0.05,
  quant.type = 7,
  optim.config = list(),
  refine.estim = FALSE,
  refinement.conf = list(),
  DQ_test = list()
)
```

Arguments

`Y`	A vector, matrix, zoo or xts object containing the univariate series.
`p`	Order of autoregressive quantile.
`q`	Order of lag Y values.

`model.type` The specification, one of the following:

- "SAV" (*Symmetric absolute value*)

$$f_t(\theta) = \beta_0 + \sum_{i=1}^{p} \beta_i f_{t-i}(\beta) + \sum_{j=1}^{q} \gamma_j |y_{t-j}|$$

- "AS" (*Asymmetric slope*)

$$f_t(\theta) = \beta_0 + \sum_{i=1}^{p} \beta_i f_{t-i}(\beta) + \sum_{j=1}^{q} \left(\gamma_{1,j}(y_{t-j})^+ + \gamma_{2,j}(y_{t-j})^-\right)$$

where $(y_{t-j})^+ = \max(y_{t-j}, 0)$ and $(y_{t-j})^- = \min(y_{t-j}, 0)$

- "INDGARCH" (*Indirect GARCH*)

$$f_t(\theta) = \left(\beta_0 + \sum_{i=1}^{p} \beta_i f_{t-i}^2(\beta) + \sum_{j=1}^{q} \gamma_j y_{t-j}^2\right)^{1/2}$$

- "I-CAV" (*Improved CAViaR*)

$$f_t(\theta) = \beta_0 + \beta_1 f_{t-1}(\theta) + (1-\beta_1)\left(\frac{\nu}{1-\gamma_1} I(y_{t-1} > 0) + \frac{\nu}{\gamma_1} I(y_{t-1} \leq 0)\right) |y_{t-1} - u|$$

where $\nu = \sqrt{\gamma_1^2 + (1-\gamma_1)^2}$, $0 < \gamma_1 < 1$ and u is the sample mean.

`tau` The quantile of interest. Set default to 0.05.

`band.hs` Logical parameter passed to quantreg::bandwidth.rq, if TRUE the Hall-Sheather bandwidth is computed, if FALSE it is computed the Bofinger bandwidth.

`jac.method` One of "Richardson" (default), "simple", or "complex". This determines the method to compute finite differences jacobian to as part of the standard errors calculations. See numDeriv::jacobian for more information.

`jac.options` A list passed to `method.args` in numDeriv::jacobian.

`sign.level` The `alpha` parameter of quantreg::bandwidth.rq function to control the level of significance for intended confidence intervals.

`quant.type` One of the types available in quantile. This is used to initialize the quantile process.

`optim.config` A list containing the optimization-related parameters, see `nloptr.print.options()` for more information. See `details` for more information.

`refine.estim` Should the result from the global optimization be refined with a gradient-based solver?

`refinement.conf`

A list containing the refinement optimization-related parameters, see `nloptr.print.options()` for more information

`DQ_test` A list containing the options for the Dynamic quantile test, the default values are `list(lags=4,constant=FALSE)` where `lags` determine the Hits lags to use, and the logical `constant` determines wheter a constant has to be included too.

Details

The main difference in this implementation is the optimization procedure used to estimate the parameters. The original paper begins with a grid search and alternates between the Nelder-Mead and quasi-Newton optimization algorithms until convergence. In this function, however, a global optimization algorithm is used: the Multilevel Single Linkage ("NLOPT_GN_MLSL_LDS"). At each iteration, the sampled points are quasi-deterministic, using Sobol's low discrepancy sequences. Thus, different optimization runs are much less dependent on the choice of random seed.

As local optimizer, it is set by default to the `NELDER-MEAD`. However, other option is `SUBPLEX` which uses the `NELDER-MEAD` is subspaces, and is claimed to be more efficient and robust than the later. One can change this as described in the nloptr::nloptr function.

An alternative to MLSL is the Improved Stochastic Ranking Evolution Strategy ("NLOPT_GN_ISRES"). However, this algorithm is much more dependent on the choice of the random seed. Although it is theoretically the case that it should return the same estimates among different runs, in practice, the user should be careful when choosing this algorithm. Furthermore, the standard errors are computed using the numerical Jacobian from `numDeriv`, and the bandwidth is computed as described in White et al. (2015).

Value

A list containing different results from the estimation. Class `CAViaR_estim`.

References

Engle, R. F., & Manganelli, S. (2004). CAViaR: Conditional Autoregressive Value at Risk by Regression Quantiles. Journal of Business & Economic Statistics, 22(4), 367–381.

Huang, D., Yu, B., Fabozzi, F. J., & Fukushima, M. (2009). CAViaR-based forecast for oil price risk. Energy Economics, 31(4), 511-518.

White, H., Kim, T. H., & Manganelli, S. (2015). VAR for VaR: Measuring tail dependence using multivariate regression quantiles. Journal of econometrics, 187(1), 169-188.

Examples

```
data=dataCAViaR
SAV <- CAViaR(Y=data$GM[1:2892],model.type = "SAV",
p=1,q=1,band.hs = TRUE,quant.type = 7,
tau=0.05,refine.estim = FALSE)

summary(SAV)
#or
SAV

plot(SAV)
```

dataCAViaR *Original caviar data used in Engle & Manganelli (2004).*

Description

Original caviar data used in Engle & Manganelli (2004).

Usage

```
dataCAViaR
```

Format

3392 rows x 3 columns of log returns of General Motors, IBM and S&P 500.

Source

https://www.simonemanganelli.org/Simone/Research.html

References

Engle, R. F., & Manganelli, S. (2004). CAViaR: Conditional autoregressive value at risk by regression quantiles. Journal of business & economic statistics, 22(4), 367-381.

MVMQ *Data used in White et al. (2015)*

Description

Data used in White et al. (2015)

Usage

```
MVMQ
```

Format

An xts object containing the returns of Goldman sachs, Braclays, HSBC and Deutsche Bank, as well as the corresponding index.

Source

https://www.simonemanganelli.org/Simone/Research.html

References

White, H., Kim, T. H., & Manganelli, S. (2015). VAR for VaR: Measuring tail dependence using multivariate regression quantiles. Journal of econometrics, 187(1), 169-188.

MVMQ_CAViaR *MVMQ-CAViaR model estimation proposed by White et al. (2015).*

Description

MVMQ-CAViaR model estimation proposed by White et al. (2015).

Usage

```
MVMQ_CAViaR(
  Y,
  p = 1,
  q = 1,
  tau = rep(0.05, ncol(Y)),
  band.hs = FALSE,
  jac.method = "simple",
  jac.options = list(),
  sign.level = 0.05,
  quant.type = 7,
  optim.config = list(),
  init.optim.config = list(),
  global.estim = FALSE,
  global.optim = list()
)
```

Arguments

Y	A matrix, xts or zoo object of multivariate series.
p	Order of autoregressive quantile.
q	Order of lag Y values.
tau	A vector containing the quantiles of interest. Set default to 0.05.
band.hs	Logical parameter passed to quantreg::bandwidth.rq, if TRUE the Hall-Sheather bandwidth is computed, if FALSE it is computed the Bofinger bandwidth.
jac.method	One of "Richardson" (default), "simple", or "complex". This determines the method to compute finite differences jacobian to as part of the standard errors calculations. See numDeriv::jacobian for more information.
jac.options	A list passed to `method.args` in numDeriv::jacobian.
sign.level	The `alpha` parameter of quantreg::bandwidth.rq function to control the level of significance for intended confidence intervals.
quant.type	One of the types available in quantile. This is used to initialize the quantile process.
optim.config	A list containing the optimization-related parameters, see `nloptr.print.options()` and `details` for more information.

`init.optim.config`
: A list containing the univariate optimization-related parameters, see `nloptr.print.options()` for more information.

`global.estim`
: Should the optimization be performed in first place with global optimization with GenSA::GenSA in order to potentially achieve a better exploration of parameter space?

`global.optim`
: A list containing the global optimization-related parameters, see GenSA::GenSA for more information.

Details

This implementation follows in essence the same optimization strategy as White et al. (2015), by first obtaining for each series their univariate CAViaR estimation with CAViaR, and setting the rest of the parameters to 0. As mentioned above, staring from those univariate estimates, this function offers two different optimization strategies: i) feed this starting point to a local optimizer (by default `NLOPT_LN_SBPLX`) following the original work, in such case the user should set `global.estim`=FALSE (the default); ii) feed this starting point to GenSA::GenSA in order perform a global optimization with the intention to explore the parameter space, and the result from this global phase is then used as starting point for the same procedure as i), then the user should set `global.estim`=TRUE.

Regarding the specification, at the moment, only the symmetric absolute value specification is available, having the following form:

$$\boldsymbol{f_t(\theta)} = \boldsymbol{c} + \sum_{i=1}^{p} \boldsymbol{A_i f_{t-i}(\theta)} + \sum_{j=1}^{q} \boldsymbol{B_j |Y_{t-j}|}$$

Value

A list containing different results from the estimation. Class `MVMQ_CAViaR`

References

White, H., Kim, T. H., & Manganelli, S. (2015). VAR for VaR: Measuring tail dependence using multivariate regression quantiles. Journal of econometrics, 187(1), 169-188.

Examples

```
Barclays <- MVMQ_CAViaR(MVMQ[,c(6,1)],tau =c(0.01,0.01),band.hs = TRUE)
summary(Barclays)
#or
Barclays
plot(Barclays,rows=2,columns=1)
```

plot.CAViaR_estim *Plot CAViaR*

Description

Plot CAViaR

Usage

```
## S3 method for class 'CAViaR_estim'
plot(x, .by = "month", .format = "%b-%Y", titl = "VaR", ...)
```

Arguments

x	CAViaR_estim class object.
.by	Frequency to display the dates in the plot, see axis.Date.
.format	Format of the displayed dates in the plot, see axis.Date.
titl	Optional title of the plot.
...	other arguments to plot

Value

No value returned

plot.MVMQ_CAViaR *Plot MVMQ_CAViaR*

Description

Plot MVMQ_CAViaR

Usage

```
## S3 method for class 'MVMQ_CAViaR'
plot(
  x,
  rows = 2,
  columns = 1,
  .by = "month",
  .format = "%b-%Y",
  titl = "VaR at ",
  ...
)
```

Arguments

x	MVMQ_CAViaR class object.
rows	Number of rows to display in the plot, passed to par
columns	Number of columns to display in the plot, passed to par
.by	Frequency to display the dates in the plot, see axis.Date.
.format	Format of the displayed dates in the plot, see axis.Date.
titl	Optional title of the plot.
...	other arguments to plot

Value

No value returned

---

print.CAViaR_estim *Print CAViaR*

Description

Print CAViaR

Usage

```
## S3 method for class 'CAViaR_estim'
print(x, digits = 5, conf.level = 0.95, ...)
```

Arguments

x	CAViaR_estim class object.
digits	number of decimals to display, passed to round.
conf.level	Confidence level to the coverage test.
...	Other arguments passed to print.

Value

No value returned

print.MVMQ_CAViaR *Print MVMQ_CAViaR*

Description

Print MVMQ_CAViaR

Usage

```
## S3 method for class 'MVMQ_CAViaR'
print(x, digits = 5, conf.level = 0.95, ...)
```

Arguments

x	MVMQ_CAViaR class object.
digits	number of decimals to display, passed to round.
conf.level	Confidence level to the coverage test.
...	Other arguments passed to print.

Value

No value returned

summary.CAViaR_estim *Summary CAViaR*

Description

Summary CAViaR

Usage

```
## S3 method for class 'CAViaR_estim'
summary(object, digits = 5, conf.level = 0.95, ...)
```

Arguments

object	CAViaR_estim class object.
digits	number of decimals to display, passed to round.
conf.level	Confidence level to the coverage test.
...	Other arguments passed to print.

Value

No value returned

`summary.MVMQ_CAViaR` *Summary MVMQ_CAViaR*

Description

Summary MVMQ_CAViaR

Usage

```
## S3 method for class 'MVMQ_CAViaR'
summary(object, digits = 5, conf.level = 0.95, ...)
```

Arguments

`object`	`MVMQ_CAViaR` class object.
`digits`	number of decimals to display ,passed to round.
`conf.level`	Confidence level to the coverage test.
`...`	Other arguments passed to print.

Value

No value returned

`wti_data` *Data used in Huang et al. (2009).*

Description

Data used in Huang et al. (2009).

Usage

```
wti_data
```

Format

An xts object of the WTI price daily returns

Source

https://fred.stlouisfed.org/series/WTISPLC

References

Huang, D., Yu, B., Fabozzi, F. J., & Fukushima, M. (2009). CAViaR-based forecast for oil price risk. Energy Economics, 31(4), 511-518.

Index

∗ **datasets**
dataCAViaR, 5
MVMQ, 5
wti_data, 11

axis.Date, *8*, *9*

CAViaR, 2, *7*

dataCAViaR, 5

FALSE, *3*, *6*

GenSA::GenSA, *7*

MVMQ, 5
MVMQ_CAViaR, 6

nloptr.print.options(), *3*, *6*, *7*
nloptr::nloptr, *4*
numDeriv::jacobian, *3*, *6*

par, *9*
plot.CAViaR_estim, 8
plot.MVMQ_CAViaR, 8
print, *9–11*
print.CAViaR_estim, 9
print.MVMQ_CAViaR, 10

quantile, *3*, *6*
quantreg::bandwidth.rq, *3*, *6*

round, *9–11*

summary.CAViaR_estim, 10
summary.MVMQ_CAViaR, 11

TRUE, *3*, *6*

wti_data, 11